

An Efficient Optimal Algorithm for Integer-Forcing Linear MIMO Receivers Design

Jinming Wen, Lanping Li, Xiaohu Tang, *Member, IEEE*, Wai Ho Mow, *Senior Member, IEEE* and Chintha Tellambura, *Fellow, IEEE*

Abstract—Although multiple-input and multiple-output (MIMO) wireless systems achieve significant data rates (e.g., multiplexing) and reliability (e.g., diversity) and are main enablers for high-rate 5G wireless systems, MIMO receivers require high implementation complexity. A solution is the use of integer-forcing (IF) linear receivers which first create an effective integer channel matrix. In this paper, we propose an efficient algorithm to find the optimal integer coefficient matrix which maximizes the achievable rate of the IF linear receiver. The algorithm initializes with a suboptimal matrix, which is updated by a novel and efficient algorithm during the process of an improved version of sphere decoding until the optimal matrix is obtained. We theoretically show that the proposed algorithm indeed finds the optimal coefficient matrix. Furthermore, theoretical complexity analysis indicates that the complexity of the new algorithm in big-O notation is an order of magnitude smaller, with respect to the dimension of the model matrix, than that of the so far most efficient algorithm. Simulation results are also presented to confirm the efficiency of our novel algorithm.

Index Terms—Integer-forcing linear receiver design, sphere decoding, successive minima problem, achievable rate.

I. INTRODUCTION

Due to the exponential growth of mobile traffic and subscribers globally, current and future wireless systems require ongoing improvements in capacity, quality and coverage. Multiple input multiple output (MIMO) uses multiple antenna arrays both in transmitters and receivers and thus exploits the space dimension to improve wireless capacity and reliability.

J. Wen is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton T6G 2V4, Canada (e-mail: jinming1@ualberta.ca)

L. Li is with the Laboratory of Information Coding and Transmission, Southwest Jiaotong University, Chengdu 610031, China (e-mail: lanping@my.swjtu.edu.cn).

X. Tang is with the Information Security and National Computing Grid Laboratory, Southwest Jiaotong University, Chengdu 610031, China (e-mail: xhutang@swjtu.edu.cn).

W.-H., Mow is with the Department of Electrical and Computer Engineering, HKUST, Hong Kong (e-mail: eewhmow@ust.hk).

C. Tellambura is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton T6G 2V4, Canada (e-mail: chintha@ece.ualberta.ca)

The two main forms of MIMO are diversity and spatial multiplexing. Diversity may be used to increase link reliability. In contrast, spatial multiplexing (SM) increases the spectral efficiency by transmitting multiple independent data streams over multiple antennas.

To achieve these gains, the MIMO receiver plays a critical role. The maximum likelihood receiver is optimal and achieves the highest rates and smallest probability of error. However, its complexity is exponential in the size of the codebook and in the number of antennas. Thus, to reduce the high computation complexity, suboptimal receivers, such as zero-forcing (ZF), successive interference cancelation (SIC) and minimum mean square error (MMSE) receivers have been developed, which achieve low-complexity but with some performance loss. Although lattice reductions (such as LLL reduction [1], SRQD [2] and V-BLAST [3]) can improve their performances (see, e.g., [4]), they may have large performance loss especially in the low signal-to-noise (SNR) regime.

Recently, a new integer-forcing (IF) linear receiver was proposed by Zhan *et al.* [5], [6]. The IF linear receiver first eliminates additive noise and then cancels interference between SM data streams. To do this, it uses a linear front-end to create an effective integer-valued channel matrix while minimizing the noise amplification. It exploits the fact that any integer linear combination of lattice points is still a lattice point. Thus, it first decodes linear combinations of the data streams and then decodes the transmit data streams. Since the coefficients of the effective channel matrix are chosen to minimize the noise amplification, the IF receiver can attain higher rates. That is, it outperforms ZF and similar receiver algorithms.

Based on the optimization criterion for IF coefficient matrix \mathbf{A} [6], Wei *et al.* [7] presented an algorithm to design \mathbf{A} based on the Fincke-Pohst search. But their algorithm does not preprocess with lattice reduction, and the choice of initial radius, which creates a set of candidate vectors for \mathbf{A} , is suboptimal, its complexity is high. Although a more efficient algorithm, which forms the optimal \mathbf{A} by exactly solving the successive minima problem based on the Schnorr-Euchner

search, is proposed in [8], highly efficient faster algorithms are still desirable. There are also some suboptimal methods which are of lower complexity at the expense of performance loss, for examples, the lattice reduction based algorithms [9] and the slowest descent method [10]. There are also some variants of IF, see, e.g., [11].

This paper focuses on developing an efficient algorithm that finds the optimal coefficient matrix \mathbf{A} for IF receiver design. Specifically, an efficient algorithm based on sphere decoding will be proposed. The main novelty of the algorithm is that it initializes with a suboptimal matrix, which is updated by a novel and efficient algorithm during the process of an improved version of sphere decoding until the optimal matrix is obtained. We will rigorously show that the new algorithm indeed finds the optimal matrix. Theoretically complexity analysis indicates that the complexity of the new algorithm in big-O notation is an order of magnitude smaller, with respect to the dimension of the model matrix, than that of the so far most efficient algorithm, which was proposed in [8]. Simulation results will also be presented to show the efficiency of the new algorithm.

The rest of the paper is organized as follows. In Section II, we introduce the coefficient matrix design problem for IF receiver design. In Section III, we introduce the lattice background material necessary to understand the current and proposed algorithms for finding \mathbf{A} . We propose our new optimal algorithm in Section V and analyze its complexity in VI. We provide a comparative performance evaluation of the proposed and existing algorithms in Section VII. Finally, conclusions are given in Section VIII.

Notation. Let \mathbb{R}^n and \mathbb{Z}^n be the spaces of the n -dimensional real and integer vectors, respectively. Let $\mathbb{R}^{m \times n}$ and $\mathbb{Z}^{m \times n}$ be the spaces of the $m \times n$ real and integer matrices, respectively. Boldface lowercase letters denote column vectors and boldface uppercase letters denote matrices. Let x_i be the i -th element of \mathbf{x} . For a matrix \mathbf{A} , let a_{ij} denote the element at row i and column j , \mathbf{a}_i be its i -th column and $\mathbf{A}_{[1,i]}$ be the submatrix of \mathbf{A} formed by its first i columns. Let $\lfloor \mathbf{x} \rfloor$ denotes the nearest integer vector of \mathbf{x} , i.e., each entry of \mathbf{x} is rounded to its nearest integer (if there is a tie, the one with smaller magnitude is chosen).

II. PROBLEM STATEMENT

In this paper, we assume a slow fading channel model where the channel remains constant over the entire block length. Since a complex MIMO system can be readily transformed to an equivalent real system, we consider the real-valued channel model only.

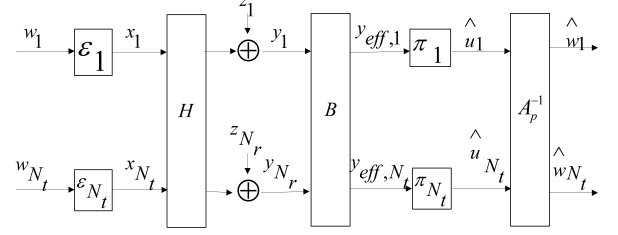


Fig. 1. The block diagram of an IF MIMO system

In the IF MIMO system (see Fig. 1), the m -th transmitter antenna is equipped with a lattice encoder ϵ_m , which maps the length- k message \mathbf{w}_m into a length- n lattice codeword $\mathbf{x}_m \in \mathbb{R}^n$, i.e.,

$$\epsilon_m : \mathbb{F}_p^k \rightarrow \mathbb{R}^n, \quad \mathbf{w}_m \rightarrow \mathbf{x}_m,$$

where the entries of \mathbf{w}_m are independently and uniformly distributed over a prime-size finite field $\mathbb{F}_p = \{0, 1, \dots, p-1\}$, i.e.,

$$\mathbf{w}_m \in \mathbb{F}_p^k, \quad m = 1, 2, \dots, N_t.$$

All transmit antennas employ the same lattice code. Each codeword satisfies the power constraint of

$$\frac{1}{n} \|\mathbf{x}_m\|^2 \leq P.$$

Let $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_{N_t}] \in \mathbb{R}^{N_t \times n}$, then the received signal $\mathbf{Y} \in \mathbb{R}^{N_r \times n}$ is given by

$$\mathbf{Y} = \mathbf{H}\mathbf{X} + \mathbf{Z}$$

where $\mathbf{H} \in \mathbb{R}^{N_r \times N_t}$ is the channel matrix, and $\mathbf{Z} \in \mathbb{R}^{N_r \times n}$ is the noise matrix. All the elements of both \mathbf{H} and \mathbf{Z} independently and identically follow the normal distribution $\mathcal{N}(0, 1)$.

The receiver aims to find a coefficient matrix $\mathbf{A} \in \mathbb{Z}^{N_t \times N_t}$ and a filter matrix $\mathbf{B} \in \mathbb{R}^{N_t \times N_r}$. With the matrix \mathbf{B} , the received message \mathbf{Y} will be projected to the more effective received vector for further decoding. The m -th filter outputs

$$\mathbf{y}_{\text{eff},m} = \mathbf{b}_m^T \mathbf{Y} = \mathbf{a}_m^T \mathbf{X} + (\mathbf{b}_m^T \mathbf{H} - \mathbf{a}_m^T) \mathbf{X} + \mathbf{b}_m^T \mathbf{Z}$$

where

$$\mathbf{z}_{\text{eff},m} = (\mathbf{b}_m^T \mathbf{H} - \mathbf{a}_m^T) \mathbf{X} + \mathbf{b}_m^T \mathbf{Z}$$

is the effective noise, \mathbf{a}_m^T and \mathbf{b}_m^T respectively denote the m -th row of \mathbf{A} and \mathbf{B} .

The receiver recovers the original N_t messages $\mathbf{W} = [\mathbf{w}_1, \dots, \mathbf{w}_{N_t}]^T$ by decoding $\mathbf{Y} = [\mathbf{y}_{\text{eff},1}, \dots, \mathbf{y}_{\text{eff},N_t}]^T$ in parallel based on the algebraic structure of lattice codes, i.e., the integer combination of lattice codewords is still a

codeword. In this way, we first recover the linear equation $\mathbf{u}_m = [\mathbf{a}_m^T \mathbf{W}] \bmod p$ from the $\mathbf{y}_{eff,m}$ at one decoder,

$$\pi_m : \mathbb{R}^n \rightarrow \mathbf{F}_p^k, \quad \mathbf{y}_{eff,m} \rightarrow \hat{\mathbf{u}}_m.$$

Indeed, the original messages can be recovered free of error as long as all the lattice equations are correctly detected, i.e.,

$$[\hat{\mathbf{w}}_1, \dots, \hat{\mathbf{w}}_{N_t}]^T = \mathbf{A}_p^{-1} [\hat{\mathbf{u}}_1, \dots, \hat{\mathbf{u}}_{N_t}]^T,$$

where $\mathbf{A}_p = [\mathbf{A}] \bmod p$ is full-rank over \mathbb{Z}_p .

Hence, the design of IF receiver is the construction of a full rank IF matrix $\mathbf{A} \in \mathbb{Z}^{N_t \times N_t}$ such that the achievable rate is maximized. For more details, see [6] and [12].

At the m -th decoder π_m , the achievable rate is,

$$\mathbf{R}_m = \frac{1}{2} \log^+ \left(\frac{P}{P \|\mathbf{H}^T \mathbf{b}_m - \mathbf{a}_m\|_2^2 + \|\mathbf{b}_m\|_2^2} \right),$$

where $\log^+(x) \triangleq \max(\log(x), 0)$. Moreover,

$$\mathbf{b}_m^T = \mathbf{a}_m^T \mathbf{H}^T (\mathbf{H} \mathbf{H}^T + \mathbf{I}/P)^{-1},$$

the achievable rate is

$$\mathbf{R}_m = \frac{1}{2} \log^+ \left(\frac{1}{\mathbf{a}_m^T \mathbf{G} \mathbf{a}_m} \right),$$

where

$$\mathbf{G} = \mathbf{I} - \mathbf{H}^T (\mathbf{H} \mathbf{H}^T + \mathbf{I}/P)^{-1} \mathbf{H}. \quad (1)$$

The total achievable rate is

$$\mathbf{R}_{total} = N_t \min \{\mathbf{R}_1, \mathbf{R}_2, \dots, \mathbf{R}_{N_t}\}.$$

In this paper, we want to maximize the total achievable rate, or equivalently, we need to solve the following optimization problem,

$$\mathbf{A}^* = \arg \min_{\substack{\mathbf{A} \in \mathbb{Z}^{N_t \times N_t} \\ \det(\mathbf{A}) \neq 0}} \max_{1 \leq m \leq N_t} \mathbf{a}_m^T \mathbf{G} \mathbf{a}_m, \quad (2)$$

where \mathbf{G} is defined in (1).

III. PRELIMINARIES OF LATTICES

This section briefly reviews the necessary background of lattice material.

A. SIVP and SMP

Let \mathbf{G} in (1) have the following Cholesky factorization:

$$\mathbf{G} = \mathbf{R}^T \mathbf{R}, \quad (3)$$

where $\mathbf{R} \in \mathbb{R}^{N_t \times N_t}$ is an upper triangular matrix. (Note that Cholesky factorization algorithms can be found in many references, see, e.g., [13].) Then, (2) can be transformed to:

$$\mathbf{A}^* = \arg \min_{\substack{\mathbf{A} \in \mathbb{Z}^{N_t \times N_t} \\ \det(\mathbf{A}) \neq 0}} \max_{1 \leq m \leq N_t} \|\mathbf{R} \mathbf{a}_m\|_2. \quad (4)$$

To solve (4), we need to find a nonsingular matrix $\mathbf{A}^* = [\mathbf{a}_1^*, \dots, \mathbf{a}_{N_t}^*] \in \mathbb{Z}^{N_t \times N_t}$ such that $\max_{1 \leq m \leq N_t} \|\mathbf{R} \mathbf{a}_m^*\|_2$ is as small as possible. In lattice theory, this value is called the N_t -th successive minimum of lattice $\mathcal{L}(\mathbf{R}) = \{\mathbf{R} \mathbf{a} | \mathbf{a} \in \mathbb{Z}^{N_t}\}$. More generally, the k -th ($1 \leq k \leq N_t$) successive minimum λ_k of $\mathcal{L}(\mathbf{R})$ is the smallest r such that the closed N_t -dimensional ball $\mathbb{B}(\mathbf{0}, r)$ of radius r centered at the origin contains k linearly independent lattice vectors. Mathematically, we have

$$\lambda_k = \min\{r | \dim(\text{span}(\mathcal{L}(\mathbf{R}) \cap \mathbb{B}(\mathbf{0}, r))) \geq k\}.$$

Thus, solving (4) is equivalent to solving a shortest independent vector problem (SIVP) which is defined as:

Definition 1. *SIVP: finding an invertible integer matrix $\mathbf{A}^* = [\mathbf{a}_1^*, \dots, \mathbf{a}_{N_t}^*] \in \mathbb{Z}^{N_t \times N_t}$ such that*

$$\max_{1 \leq i \leq N_t} \|\mathbf{R} \mathbf{a}_i^*\| \leq \lambda_{N_t},$$

where λ_{N_t} is the N_t -th successive minima.

As pointed out in [8], instead of solving the SIVP, it is advantageous to solve a successive minima problem (SMP) which is defined as:

Definition 2. *SMP: finding an invertible integer matrix $\mathbf{A}^* = [\mathbf{a}_1^*, \dots, \mathbf{a}_{N_t}^*] \in \mathbb{Z}^{N_t \times N_t}$ such that*

$$\|\mathbf{R} \mathbf{a}_i^*\| = \lambda_i, \quad 1 \leq i \leq N_t.$$

In addition to integer-forcing, solving a SMP is needed in many other applications including physical-layer network coding [14], and the expanded compute-and-forward framework [15].

In this paper, we will focus on how to solve the SMP. Clearly, the SMP is a variant of the shortest vector problem (SVP) which is defined as:

$$\mathbf{a}^* = \min_{\mathbf{a} \in \mathbb{Z}^{N_t} \setminus \{\mathbf{0}\}} \|\mathbf{R} \mathbf{a}\|_2^2. \quad (5)$$

SVP arises from enormous applications including in communications (see, e.g., [16], [17], [18]) and cryptography (see, e.g., [19], [20], [21]).

Both of the methods proposed in [7] and [8] for solving (4) actually solve the SMP by modifying the algorithms for solving (5). Thus, it is necessary to introduce one of its frequently used solvers, i.e., the discrete search approach. This approach is referred to as sphere decoding in communications, such as the Schnorr-Euchner algorithm [22], a variation of the Fincke-Pohst search strategy [23], which enumerates the integer vector in a hyper-sphere. To make the search faster, a lattice reduction is often performed to transform the given problem to an equivalent but easier solved problem. One of the commonly used reductions is the LLL reduction proposed in [1]. We respectively present the LLL reduction and the Schnorr-Euchner algorithm in Sec. III-B and III-C.

B. LLL reduction

After the Cholesky factorization of \mathbf{G} , the LLL reduction [1] reduces \mathbf{R} in (3) to $\bar{\mathbf{R}}$ through the QRZ factorization:

$$\bar{\mathbf{Q}}^T \mathbf{R} \mathbf{Z} = \bar{\mathbf{R}}, \quad (6)$$

where $\bar{\mathbf{Q}} \in \mathbb{R}^{N_t \times N_t}$ is orthogonal, $\mathbf{Z} \in \mathbb{Z}^{N_t \times N_t}$ is unimodular (i.e., \mathbf{Z} also satisfies $|\det(\mathbf{Z})| = 1$) and $\bar{\mathbf{R}} \in \mathbb{R}^{N_t \times N_t}$ is upper triangular and satisfies the following conditions:

$$\begin{aligned} |\bar{r}_{ik}| &\leq \frac{1}{2} |\bar{r}_{ii}|, \quad i = 1, 2, \dots, k-1, \\ \delta \bar{r}_{k-1,k-1}^2 &\leq \bar{r}_{k-1,k}^2 + \bar{r}_{kk}^2, \quad k = 2, 3, \dots, N_t, \end{aligned}$$

where δ is a constant satisfying $1/4 < \delta \leq 1$. The matrix $\bar{\mathbf{R}}$ is said to be LLL reduced.

The LLL reduction algorithm can be found in [1] and its properties have been studied in [4].

After the LLL reduction (6), SVP (5) is then transformed to:

$$\mathbf{b}^* = \min_{\mathbf{b} \in \mathbb{Z}^{N_t} \setminus \{0\}} \|\bar{\mathbf{R}}\mathbf{b}\|_2. \quad (7)$$

Moreover \mathbf{a}^* and \mathbf{b}^* satisfy $\mathbf{a}^* = \mathbf{Z}\mathbf{b}^*$.

Similarly, after the LLL reduction (6), the SMP can be transformed to the following reduced SMP (RSMP):

Definition 3. *RSMP: finding an invertible integer matrix $\mathbf{B}^* = [\mathbf{b}_1^*, \dots, \mathbf{b}_n^*] \in \mathbb{Z}^{N_t \times N_t}$ such that*

$$\|\bar{\mathbf{R}}\mathbf{b}_i^*\| = \lambda_i, \quad 1 \leq i \leq N_t.$$

And \mathbf{A}^* and \mathbf{B}^* satisfy $\mathbf{A}^* = \mathbf{Z}\mathbf{B}^*$.

C. Schnorr-Euchner search algorithm for solving SVP

In the following, we introduce the Schnorr-Euchner search algorithm to solve the SVP (7). Let the optimal solution be within the following hyper-ellipsoid

$$\|\bar{\mathbf{R}}\mathbf{b}\|_2^2 < \beta^2, \quad (8)$$

where β is a constant. Define

$$d_{N_t} = 0, \quad d_k = -\frac{1}{\bar{r}_{kk}} \sum_{j=k+1}^{N_t} \bar{r}_{kj} b_j, \quad k = N_t - 1, \dots, 1. \quad (9)$$

Then (8) can be rewritten as

$$\sum_{i=1}^{N_t} \bar{r}_{ii}^2 (b_i - d_i)^2 < \beta^2$$

which is equivalent to

$$\bar{r}_{kk}^2 (b_k - d_k)^2 < \beta^2 - \sum_{j=k+1}^{N_t} \bar{r}_{jj}^2 (b_j - d_j)^2 \quad (10)$$

for $k = N_t, N_t - 1, \dots, 1$, where k is called the level index and $\sum_{j=N_t+1}^{N_t} = 0$.

Based on (10), the Schnorr-Euchner search algorithm can be described as follows. First, we set the initial $\beta = \infty$, and for $k = N_t, N_t - 1, \dots, 1$, we compute d_k by (9) and set $b_k = \lfloor d_k \rfloor$, leading to $b_k = 0$, for which (10) holds. As a result, we obtain an integer vector $\mathbf{b} = \mathbf{0}$. Since $\mathbf{b}^* \neq \mathbf{0}$, we cannot set $\mathbf{b}^* = \mathbf{b}$, hence we need to update \mathbf{b} . Specifically, we set b_1 as the next closest integer to d_1 . Note that (10) with $k = 1$ holds for the updated \mathbf{b} . Then, we store this updated \mathbf{b} and set $\beta = \|\bar{\mathbf{R}}\mathbf{b}\|_2$. After this, we try to find an integer vector within the new ellipsoid by updating the latest found \mathbf{b} . Obviously, we cannot update only its first entry b_1 , since we cannot find any new integer b_1 that satisfies (10) with $k = 1$, which is now an equality for the current \mathbf{b} . Thus we move up to level 2 to try to update b_2 by choosing it being the next nearest integer to d_2 . If it satisfies (10) with $k = 2$, we move down to level 1 to update b_1 by computing d_1 (see (9)) and setting $b_1 = \lfloor d_1 \rfloor$ and then checking if (10) with $k = 1$ holds and so on; Otherwise we move up to level 3 to try to update b_3 , and so on. Finally, when we fail to find a new value for b_{N_t} to satisfy (10) with $k = N_t$, the search process stops and the latest found integer vector is \mathbf{b}^* we seek. For more details on this algorithm, see, e.g., [17] and [24]. There are enormous variants of SE, see, e.g., [25], [26], [27], [28].

IV. EXISTING METHODS

A. The optimal method of finding \mathbf{A}^* in [7]

The optimal method of finding \mathbf{A}^* in [7] consists of two steps. In the first step, a set Ω is used to store all the vectors \mathbf{a} satisfying

$$\|\mathbf{R}\mathbf{a}\|_2^2 \leq \beta, \quad (11)$$

with $\beta = \mathbf{u}^T \mathbf{G} \mathbf{u}$, where

$$u_i = \begin{cases} 1, & v_i \geq 0 \\ -1, & v_i < 0 \end{cases}, \quad 1 \leq i \leq N_t$$

with \mathbf{v} being the eigenvector corresponding to the minimal eigenvalue of \mathbf{G} (see (1)). Set Ω is obtained by enumerating all the vectors satisfying (11) with the Fincke-Pohst search algorithm. Since LLL reduction is not used to reduce \mathbf{R} (see (3)), β can be very large, so does $|\Omega|$ ($|\Omega|$ denotes the number of vectors contained in Ω) especially when the dimension is large.

In the second step, all the vectors $\mathbf{a} \in \Omega$ are first sorted in a nondecreasing order based on $\mathbf{a}^T \mathbf{G} \mathbf{a}$. For the convenience of explanation, we use $\mathbf{a}_{[k]}$ to denote the k -th vector of the ordered set Ω . Then, check whether $|\Omega| \geq N_t$ holds. If it does not hold, the vectors in Ω cannot form a $N_t \times N_t$ nonsingular matrix \mathbf{A} , then set $\beta = 2\beta$ and regenerate a set Ω . Otherwise, i.e., $|\Omega| \geq N_t$, then set $k = N_t$, and search all the arbitrary $N_t - 1$ vectors contained in the set formed by the first $k - 1$ vectors of $|\Omega|$ to see whether these vectors together with $\mathbf{a}_{[k]}$ can be formed a $N_t \times N_t$ nonsingular matrix \mathbf{A} or not. If yes, \mathbf{A}^* is formed by these vectors, and the algorithm terminates. Otherwise, let $k = k + 1$ and check whether this kind of \mathbf{A} can be found or not. If \mathbf{A} cannot be found even when $k = |\Omega|$, then set $\beta = 2\beta$ and regenerate a set Ω , and so on.

If β is large, $|\Omega|$ is very large, then the whole algorithm is time-consuming. On the other hand, if β is small, then one may need to set $\beta = 2\beta$ (maybe several times), the Fincke-Pohst search algorithm maybe used several times, so the complexity is also high.

B. The optimal method of finding \mathbf{A}^* in [8]

The optimal method of finding \mathbf{A}^* in [8] first uses the LLL reduction to reduce \mathbf{R} to $\bar{\mathbf{R}}$ (see (6)) and then solve the RSMP, which is defined in Definition 3, to get \mathbf{B}^* . After obtaining \mathbf{B}^* , the solution \mathbf{A}^* of the SMP is obtained by setting $\mathbf{A}^* = \mathbf{Z}\mathbf{B}^*$. The optimal solution \mathbf{B}^* is obtained column by column in N_t iterations. To be more concrete, the solution of (7) forms the first column of \mathbf{B}^* . In the k -th iteration ($2 \leq k \leq N_t$), the Schnorr-Euchner algorithm was modified to find an integer vector \mathbf{b} which not only satisfies (7) but also being independent

with the first $k - 1$ columns of \mathbf{B}^* (note that this problem is called a subspace avoiding problem in [29], [30]), and this \mathbf{b} forms the k -th column of \mathbf{B}^* . To speed up the process of solving the corresponding SVPs, in the N_t -th level, only the nonnegative entries are searched in [8]. Moreover, the search range of the first level was reduced by using the fact that

$$\|\mathbf{R}\mathbf{a}_k\| \geq \|\mathbf{R}\mathbf{a}_{k-1}\|, \quad 2 \leq k \leq N_t.$$

And some intermediate integer vectors are stored to reduce the initial radius β of the SVPs. More details on this algorithm are referred to [8].

V. A NOVEL ALGORITHM FOR SOLVING THE RSMP

In this section, we will propose a novel algorithm to solve the RSMP (see Definition 3) to get \mathbf{B}^* . Note that after obtaining \mathbf{B}^* , the solution \mathbf{A}^* of the SMP can be easily obtained by setting $\mathbf{A}^* = \mathbf{Z}\mathbf{B}^*$, thus we only need to solve the RSMP. Different from [8] which finds \mathbf{B}^* column by column, our new algorithm starts with a suboptimal solution and then update it, during the process of Schnorr-Euchner enumeration, until \mathbf{B}^* is obtained.

We first introduce some useful theorems and an efficient algorithm to update the suboptimal solution in Sec V-A. Then, we develop a novel algorithm for solving the RSMP in Sec V-B. And finally, we theoretically prove that the proposed algorithm indeed solves the RSMP in Sec V-C.

A. Preliminaries

Some useful theorems and an efficient algorithm to update the suboptimal solution will be provided in this subsection.

The following theorem shows that for each given nonsingular matrix \mathbf{B} and nonzero vector \mathbf{b} , there always exists at least one index j such that the matrix obtained by replacing \mathbf{b}_j by \mathbf{b} is still singular.

Theorem 1. Let $\mathbf{B} \in \mathbb{R}^{n \times n}$ be an arbitrary invertible matrix and $\mathbf{b} \in \mathbb{R}^n$ be an arbitrary nonzero vector such that $\tilde{\mathbf{B}}_{[1, i+1]}$ is full column rank for some i with $0 \leq i \leq n - 1$, where

$$\tilde{\mathbf{B}} = [\mathbf{b}_1 \quad \dots \quad \mathbf{b}_i \quad \mathbf{b} \quad \mathbf{b}_{i+1} \quad \dots \quad \mathbf{b}_n]. \quad (12)$$

Then there exists at least one j with $i+2 \leq j \leq n+1$ such that $\tilde{\mathbf{B}}_{[n, j]}$ is also invertible, where $\tilde{\mathbf{B}}_{[n, j]}$ is the matrix obtained by removing $\tilde{\mathbf{b}}_j$ from $\tilde{\mathbf{B}}$.

Proof: See Appendix A. ■

Note that if $i = 0$, $\tilde{\mathbf{B}}_{[1, i+1]}$ reduces to \mathbf{b} which is full column rank, and Theorem 1 reduces to:

Corollary 1. Let $\mathbf{B} \in \mathbb{R}^{n \times n}$ be an arbitrary invertible matrix and $\mathbf{b} \in \mathbb{R}^n$ be an arbitrary nonzero vector, then there exists

at least one j with $1 \leq j \leq n$ such that the matrix obtained by removing \mathbf{b}_j from $[\mathbf{b} \ \mathbf{b}_1 \ \dots \ \mathbf{b}_n]$ is also invertible.

As will be seen in the next subsection, to efficiently solve the RSMP, we need to develop a fast algorithm for the following problem: for any given nonsingular matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$ and nonzero vector $\mathbf{b} \in \mathbb{R}^n$ that satisfy

$$\|\mathbf{b}_1\|_2 \leq \|\mathbf{b}_2\|_2 \leq \dots \leq \|\mathbf{b}_n\|_2 \text{ and } \|\mathbf{b}\|_2 < \|\mathbf{b}_n\|_2, \quad (13)$$

we need to replace a column with the largest ℓ_2 -norm of \mathbf{B} by \mathbf{b} such that the resulting matrix is invertible and its columns are ordered in a nondecreasing order according to their ℓ_2 -norms.

By Theorem 1 and Corollary 1, there exists at least one j with $1 \leq j \leq n$ such that the matrix obtained by replacing \mathbf{b}_j with \mathbf{b} is still invertible, so the problem is well defined.

In the following, we develop an algorithm with complexity of $\mathcal{O}(n^3)$ flops to address it (note that, as will be explained in the following, the trivial method requires $\mathcal{O}(n^4)$ flops).

By (13), one can see that there exists i with $0 \leq i \leq n-1$ such that $\|\mathbf{b}_i\|_2 \leq \|\mathbf{b}\|_2 \leq \|\mathbf{b}_{i+1}\|_2$ (Note that if $i = 0$, it means $\|\mathbf{b}\|_2 < \|\mathbf{b}_1\|_2$). Then, one can see that

$$\|\tilde{\mathbf{b}}_1\|_2 \leq \|\tilde{\mathbf{b}}_2\|_2 \leq \dots \leq \|\tilde{\mathbf{b}}_{n+1}\|_2, \quad (14)$$

where $\tilde{\mathbf{B}}$ is defined in (12). And hence, the question is equivalent to find the largest j with $1 \leq j \leq n+1$ such that $\tilde{\mathbf{B}}_{[\setminus j]}$ is invertible (note that by (14), the columns of $\tilde{\mathbf{B}}_{[\setminus j]}$ are ordered in a nondecreasing order according to their ℓ_2 -norms).

If $\tilde{\mathbf{B}}_{[1,i+1]}$ is not full column rank, then only \mathbf{b}_{i+1} , i.e., \mathbf{b} , can be removed from $\tilde{\mathbf{B}}$, and the resulting matrix is \mathbf{B} which is invertible by assumption. This is because, no matter which $\tilde{\mathbf{b}}_j$ is removed for $i+2 \leq j \leq n+1$, the resulting matrix contains $\tilde{\mathbf{B}}_{[1,i+1]}$ as a submatrix, and hence it is not invertible. And if $\tilde{\mathbf{b}}_j$ is removed for $1 \leq j \leq i$, then it is not the column with the largest index being reduced.

In the following, we assume $\tilde{\mathbf{B}}_{[1,i+1]}$ is full column rank, then by Theorem 1, there exists at least one j with $i+2 \leq j \leq n+1$ such that $\tilde{\mathbf{B}}_{[\setminus j]}$ is invertible. A natural method to find the desire j is to check whether $\tilde{\mathbf{B}}_{[\setminus j]}$ is invertible for $j = n+1, n, \dots, i+1$ until an invertible matrix is found. Clearly, this approach works, but the main drawback of this method is its worst complexity is $\mathcal{O}(n^4)$ flops which is too high. Concretely, in the worst case, i.e., in the case that only $\tilde{\mathbf{B}}_{[\setminus 1]}$ is invertible, then $n+1$ matrices are needed to be checked. Since the complexity of checking whether a $n \times n$ matrix is invertible or not is $\mathcal{O}(n^3)$ flops, the whole complexity is $\mathcal{O}(n^4)$ flops.

In the following, we introduce a method which can find j in $\mathcal{O}(n^3)$ flops. We start with introducing the following theorem

which shows that if the matrix formed by a given nonzero vector \mathbf{b} and the first j columns of a given nonsingular matrix \mathbf{B} is not full column rank, then $\tilde{\mathbf{B}}_{[\setminus j]}$ is invertible.

Theorem 2. Let $\mathbf{B} \in \mathbb{R}^{n \times n}$ be an arbitrary invertible matrix and $\mathbf{b} \in \mathbb{R}^n$ be an arbitrary nonzero vector such that $\tilde{\mathbf{B}}_{[1,i+1]}$ (see (12)) is full column rank for some i with $0 \leq i \leq n-1$. If there exists some j with $i+2 \leq j \leq n$ such that $\tilde{\mathbf{B}}_{[1,j]}$ is not full column rank, then $\tilde{\mathbf{B}}_{[\setminus j]}$ is invertible.

Proof: We prove it by contradiction. We assume $\tilde{\mathbf{B}}_{[\setminus j]}$ is not invertible, and show that \mathbf{B} is not invertible either, which is contradict with the assumption.

Since $\tilde{\mathbf{B}}_{[\setminus j]}$ is not invertible, $\tilde{\mathbf{b}}_{j+1}$ (i.e., \mathbf{b}_j) is a linear combination of vectors

$$\{\tilde{\mathbf{b}}_k | 1 \leq k \leq n+1, k \neq j, j+1\},$$

which is actually

$$\{\mathbf{b}\} \cup \{\mathbf{b}_k | 1 \leq k \leq n, k \neq j-1, j\}.$$

Similarly, as $\tilde{\mathbf{B}}_{[1,j]}$ is not full column rank, \mathbf{b} is a linear combination of vectors

$$\{\mathbf{b}_k | 1 \leq k \leq j-1\}.$$

Thus, \mathbf{b}_j is a linear combination of vectors

$$\{\mathbf{b}_j | 1 \leq j \leq n, k \neq j\},$$

and this leads to that \mathbf{B} is not invertible, which is contradict with the assumption that \mathbf{B} is invertible. Thus, the assumption is not correct. In other words, $\tilde{\mathbf{B}}_{[\setminus j]}$ is invertible. ■

Similar to Theorem 1, when $i = 0$, Theorem 2 reduces to:

Corollary 2. Let $\mathbf{B} \in \mathbb{Z}^{n \times n}$ be an arbitrary invertible matrix and $\mathbf{b} \in \mathbb{Z}^n$ be an arbitrary nonzero vector such that $\{\mathbf{b}, \mathbf{b}_1, \dots, \mathbf{b}_j\}$ is linearly dependent for some j with $1 \leq j \leq n-1$, then the matrix obtained by removing \mathbf{b}_j from $[\mathbf{b} \ \mathbf{b}_1 \ \dots \ \mathbf{b}_n]$ is invertible.

The following theorem shows how to find the desire j .

Theorem 3. Let $\mathbf{B} \in \mathbb{R}^{n \times n}$ be an arbitrary invertible matrix and $\mathbf{b} \in \mathbb{R}^n$ be an arbitrary nonzero vector such that $\tilde{\mathbf{B}}_{[1,i+1]}$ (see (12)) is full column rank for some i with $0 \leq i \leq n-1$. Suppose that j is the smallest integer with $i+1 < j \leq n$ such that $\tilde{\mathbf{B}}_{[1,j]}$ is not full column rank, then j is the largest integer with $j \leq n$ such that $\tilde{\mathbf{B}}_{[\setminus j]}$ is invertible.

Proof: By Theorem 2, $\tilde{\mathbf{B}}_{[\setminus j]}$ is invertible. In the following, we show that $\tilde{\mathbf{B}}_{[\setminus k]}$ is not invertible for any k with $j < k \leq n+1$ by contradiction.

Suppose that there exists k with $j < k \leq n+1$ such that $\tilde{\mathbf{B}}_{[\setminus k]}$ is invertible. Then $\tilde{\mathbf{B}}_{[1,j]}$ is full column rank which is contradict with the assumption. Thus, the assumption is wrong. In other words, j is the largest integer with $j \leq n$ such that $\tilde{\mathbf{B}}_{[\setminus j]}$ is invertible. ■

Remark 1. In Theorem 3, we assumed $j \leq n$, this is because if $\tilde{\mathbf{B}}_{[1,j]}$ is full column rank for all j with $i+1 < j \leq n$, then $\tilde{\mathbf{B}}_{[\setminus(n+1)]}$ is invertible, and hence we can set $j = n+1$.

By Theorem 3, one of the methods to find j is to check whether $\tilde{\mathbf{B}}_{[1,j]}$ is full column rank for $j = i+1, \dots, n$ until find a j such that it is not full column rank. If $\tilde{\mathbf{B}}_{[1,j]}$ is full column rank for all $i+1 \leq j \leq n$, then set $j = n+1$. But again, the worst complexity is $\mathcal{O}(n^4)$ flops. Concretely, when $i = 1$ and $j = n$, then $\tilde{\mathbf{B}}_{[1,j]}$ should be checked for all $2 \leq j \leq n$, and hence the complexity is $\mathcal{O}(n^4)$ flops.

In the following, we develop an algorithm to find j with the complexity of $\mathcal{O}(n^3)$ flops. To clearly explain the algorithm, we need to introduce the definition of the row echelon form for matrices which can be found in many linear algebra books (see, e.g., [31, p.39]).

Definition 4. A matrix is in row echelon form if it satisfies the following two conditions:

- 1) All nonzero rows (rows with at least one nonzero element) are above any rows of all zeroes (all zero rows, if any, belong at the bottom of the matrix);
- 2) The leading coefficient (the first nonzero number from the left, also called the pivot) of a nonzero row is always strictly to the right of the leading coefficient of the row above it.

Note that in some textbooks (see, e.g., [32, p.11]), the leading coefficient is required to be 1 for the definition of row echelon form.

The main reason for introducing the row echelon form for a matrix is whether its submatrices are full column rank or not can be easily checked. As a result, the specific j we want can also be easily found. Specifically, we have the following algorithm for the problem raised in the paragraph under Corollary 1.

By Theorem 3, one can easily see that this algorithm indeed works. Moreover, the complexity of the second step, which dominates the whole algorithm, is $\mathcal{O}(n^3)$ stops via Gaussian elimination [31, p.44], so the total complexity is $\mathcal{O}(n^3)$ flops.

B. A novel algorithm for solving the RSMP

In this subsection, we develop a novel and efficient algorithm for solving the RSMP based on the Schnorr-Euchner search algorithm and Algorithm 1.

Algorithm 1 Efficient algorithm for updating \mathbf{B}

Input: A full column rank matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$ and a nonzero vector $\mathbf{b} \in \mathbb{R}^n$ that satisfy (13).

Output: $\tilde{\mathbf{B}}$ which is obtained by replacing a column with the largest ℓ_2 -norm of \mathbf{B} with \mathbf{b} such that the resulting matrix is still invertible and its columns are ordered in a nondecreasing order according to their ℓ_2 -norms.

- 1: Find i and form $\tilde{\mathbf{B}}$ (see (12)) such that it satisfies (14);
 - 2: Reduce $\tilde{\mathbf{B}}$ into its row echelon form by Gaussian elimination, for more details, see, e.g., [31, pp.40-41], and denote it by $\tilde{\mathbf{R}}$.
 - 3: Check whether $\tilde{r}_{jj} = 0$ for $j = i+1, \dots, n$ until finding a j such $\tilde{b}_{jj} = 0$ if it exists, and set $\tilde{\mathbf{B}} = \tilde{\mathbf{B}}_{[\setminus j]}$. Otherwise, i.e., $\tilde{r}_{jj} \neq 0$ for all $i+1 \leq j \leq n$, then set $\tilde{\mathbf{B}} = \tilde{\mathbf{B}}_{[\setminus(n+1)]}$.
-

The proposed algorithm is described as follows: we start with a suboptimal solution \mathbf{B} , and specifically, we assume that this initial \mathbf{B} is a permutation of the $N_t \times N_t$ identity matrix such that

$$\|\tilde{\mathbf{R}}\mathbf{b}_1\|_2 \leq \|\tilde{\mathbf{R}}\mathbf{b}_2\|_2 \leq \dots \leq \|\tilde{\mathbf{R}}\mathbf{b}_{N_t}\|_2, \quad (15)$$

and assume $\beta = \|\tilde{\mathbf{R}}\mathbf{b}_{N_t}\|$. Then we modify the Schnorr-Euchner search algorithm to search the nonzero integer vectors \mathbf{b} satisfying (8) to update \mathbf{B} . Specifically, whenever a zero vector \mathbf{b} is obtained, we update \mathbf{b} by setting \mathbf{b}_1 as the next closest integer to d_1 to obtain a nonzero integer vector \mathbf{b} . And as long as a nonzero integer vector \mathbf{b} is obtained (note that \mathbf{b} satisfies $\|\tilde{\mathbf{R}}\mathbf{b}\|_2 < \|\tilde{\mathbf{R}}\mathbf{b}_{N_t}\|_2$), we use it to update \mathbf{B} and β . Concretely, we first use \mathbf{b} to replace \mathbf{b}_j for the largest j with $1 \leq j \leq N_t$ such that the resulting matrix (which is also denoted it by \mathbf{B}) is still invertible and satisfies (15) by Algorithm 1 (note that $\tilde{\mathbf{R}}\mathbf{B}$ and $\tilde{\mathbf{R}}\mathbf{b}$ are respectively viewed as \mathbf{B} and \mathbf{b} when using Algorithm 1). Then, we set $\beta = \|\tilde{\mathbf{R}}\mathbf{b}_n\|$. Note that β decreases only when $j = N_t$, i.e., the last column of the original \mathbf{B} is replaced by \mathbf{b} . After this, we try to update \mathbf{b} to get another nonzero integer vector to update \mathbf{B} (for more details on how to update \mathbf{b} , see Sec. III-C). Finally, when \mathbf{B} cannot be updated anymore and β cannot be decreased anymore (i.e., when we fail to find a new value for \mathbf{b}_{N_t} to satisfy (10) with $k = N_t$), the search process stops and output \mathbf{B}^* .

For efficiency, $\|\tilde{\mathbf{R}}\mathbf{b}_i\|_2, 1 \leq i \leq n$ can be stored with a vector and $\|\tilde{\mathbf{R}}\mathbf{b}\|_2$ can be calculated while using the Schnorr-Euchner enumeration strategy. Then, whenever we updating \mathbf{B} , we only need to update the vector without calculating any $\|\tilde{\mathbf{R}}\mathbf{b}_i\|_2, 1 \leq i \leq n$.

In the following, we utilize the strategy proposed in [33] to

further speed up the above process. Clearly, if \mathbf{B}^* is a solution to the RSMP, so is $\bar{\mathbf{B}}^*$, where $\bar{\mathbf{B}}^* = \mathbf{B}^*$ except that $\bar{\mathbf{b}}_j^* = -\mathbf{b}_j^*$ for any j with $1 \leq j \leq N_t$. Thus, to further speed up the above process, we only need to search the nonzero integer vectors \mathbf{b} , with $b_{N_t} \geq 0$, and $b_k \geq 0$ for $1 \leq k \leq N_t - 1$ if $b_j = 0$ for all $k + 1 \leq j \leq N_t$, to update \mathbf{B} in the above process. Note that only the former property of \mathbf{b} is exploited in [8], whereas our strategy can prune more vectors while retaining optimality.

By the above analysis, the proposed algorithm for solving the RSMP can be summarized in Algorithm 2, where

$$\text{sgn}(x) = \begin{cases} 1, & x \geq 0 \\ -1, & x < 0 \end{cases}. \quad (16)$$

Algorithm 2 A novel algorithm for solving the RSMP

Input: A nonsingular upper triangular $\bar{\mathbf{R}} \in \mathbb{R}^{N_t \times N_t}$.

Output: A solution \mathbf{B}^* to the RSMP, i.e.,

$$\|\bar{\mathbf{R}}\mathbf{b}_i^*\|_2 = \lambda_i, \quad 1 \leq i \leq N_t,$$

where λ_i is the i -th successive minima of lattice $\mathcal{L}(\bar{\mathbf{R}})$.

- 1) (Initialization) Set $k = N_t$, \mathbf{B} is the permutation of the $N_t \times N_t$ identity matrix such that (15) holds, $\beta = \|\bar{\mathbf{R}}\mathbf{b}_n\|_2$.
 - 2) Compute d_k by using (9), set $b_k = \lfloor d_k \rfloor$ and $s_k = \text{sgn}(d_k - b_k)$ (see (16)).
 - 3) (Main Step) If the inequality in (10) does not hold, then go to Step 4. Else if $k > 1$, set $k = k - 1$ and go to Step 2. Else ($k = 1$), go to Step 5.
 - 4) (Outside ellipsoid) If $k = N_t$, set $\mathbf{B}^* = \mathbf{B}$ and terminate. Else, set $k = k + 1$ and go to Step 6.
 - 5) (A valid point is found) If \mathbf{b} is a nonzero vector, use Algorithm 1 to update \mathbf{B} , set $\beta = \|\bar{\mathbf{R}}\mathbf{b}_n\|_2$ and $k = k + 1$.
 - 6) (Enumeration at level k) If $k = N_t$ or $\mathbf{b}_{k+1:N_t} = \mathbf{0}$, set $b_k = b_k + 1$. Otherwise, set $b_k = b_k + s_k$, $s_k = -s_k - \text{sgn}(s_k)$ and go to Step 3.
-

C. Proof of optimality

In this subsection, we show that the matrix obtained by Algorithm 2 indeed solves the RSMP. Specifically, we have the following theorem:

Theorem 4. Suppose that $\mathbf{B}^* \in \mathbb{Z}^{N_t \times N_t}$ is the invertible matrix returned by Algorithm 2, then

$$\|\bar{\mathbf{R}}\mathbf{b}_i^*\|_2 = \lambda_i, \quad 1 \leq i \leq N_t, \quad (17)$$

where $\bar{\mathbf{R}}$ is defined in (6) and λ_i is the i -th successive minima of lattice $\mathcal{L}(\bar{\mathbf{R}})$.

Proof: We prove the theorem by contradiction. Suppose that (17) does not hold for at least one i , and let j with $1 \leq j \leq N_t$ be the smallest integer such that $\|\bar{\mathbf{R}}\mathbf{b}_j^*\| \neq \lambda_j$. Furthermore, we assume $\bar{\mathbf{B}} \in \mathbb{Z}^{N_t \times N_t}$ is a solver of the RSMP, i.e., $\bar{\mathbf{B}} \in \mathbb{Z}^{N_t \times N_t}$ is invertible and satisfies

$$\|\bar{\mathbf{R}}\bar{\mathbf{b}}_i\|_2 = \lambda_i, \quad 1 \leq i \leq N_t.$$

Then, by the definition of λ_j , $\|\bar{\mathbf{R}}\mathbf{b}_j^*\| > \lambda_j$. Applying the aforementioned equation yields

$$\|\bar{\mathbf{R}}\bar{\mathbf{b}}_k\|_2 < \|\bar{\mathbf{R}}\mathbf{b}_j^*\|_2, \quad 1 \leq k \leq j. \quad (18)$$

In the following, we show that \mathbf{B}^* cannot be returned by Algorithm 2 which is contradict with the assumption.

We first show that there exists at least one k with $1 \leq k \leq j$ such that $[\mathbf{b}_1^* \dots \mathbf{b}_{j-1}^* \bar{\mathbf{b}}_k]$ is full column rank by contradiction. As when $j = 1$, $[\mathbf{b}_1^* \dots \mathbf{b}_{j-1}^* \bar{\mathbf{b}}_k]$ reduces to $\bar{\mathbf{b}}_k$ which is full column rank since $\bar{\mathbf{b}}_k \neq \mathbf{0}$, we only need to show it is still full column rank for $j > 1$. Suppose that $[\mathbf{b}_1^* \dots \mathbf{b}_{j-1}^* \bar{\mathbf{b}}_k]$ is not full column rank for each $1 \leq k \leq j$, then $\bar{\mathbf{b}}_k$ is a linear combination of $\{\mathbf{b}_1^*, \dots, \mathbf{b}_{j-1}^*\}$ for all $1 \leq k \leq j$ which is impossible since j linear independent vectors, i.e., $\bar{\mathbf{b}}_1, \dots, \bar{\mathbf{b}}_j$ (note that $\bar{\mathbf{B}}$ is invertible, so they are linearly independent), cannot be expressed as linear combinations of $j-1$ vectors. Thus, there exists at least one k with $1 \leq k \leq j$ such that $[\mathbf{b}_1^* \dots \mathbf{b}_{j-1}^* \bar{\mathbf{b}}_k]$ is full column rank.

Suppose that $[\mathbf{b}_1^* \dots \mathbf{b}_{j-1}^* \bar{\mathbf{b}}_k]$ is full column rank for some $1 \leq k \leq j$. Since \mathbf{B}^* is the invertible matrix returned by Algorithm 2, it satisfies (15). So by (18), $\bar{\mathbf{b}}_k$ satisfies (8) with $\beta = \|\bar{\mathbf{R}}\mathbf{b}_{N_t}^*\|$. Let $\|\bar{\mathbf{R}}\mathbf{b}_{\ell-1}^*\|_2 \leq \|\bar{\mathbf{R}}\bar{\mathbf{b}}_k\|_2 \leq \|\bar{\mathbf{R}}\mathbf{b}_{\ell}^*\|_2$ for some $1 \leq \ell \leq j$ (note that since \mathbf{B}^* satisfies (15) and (18), ℓ exists). Then, $\bar{\mathbf{R}}\bar{\mathbf{B}}$ is already in a non-decreasing order according to their ℓ_2 -norms, where $\tilde{\mathbf{B}} = [\mathbf{b}_1^* \dots \mathbf{b}_{\ell-1}^* \bar{\mathbf{b}}_k \mathbf{b}_{\ell}^* \dots \mathbf{b}_{N_t}^*]$. Then, by the process of Algorithm 2, \mathbf{B}^* will be updated to $\tilde{\mathbf{B}}_{[\ell:i]}$ for some i with $\ell + 1 \leq i \leq N_t + 1$, and the latter cannot be changed back to \mathbf{B}^* . Thus, \mathbf{B}^* cannot be returned by Algorithm 2 which is contradict with the assumption, so the theorem holds. ■

VI. COMPLEXITY ANALYSIS

In this section, we first analyze the complexity of the proposed algorithm for solving the SMP, and then show that the complexity of the new algorithm in big-O notation is an order of magnitude smaller than that of [8, Alg. 2] with respect to N_t .

A. Complexity Analysis of the Proposed Algorithm

In this subsection, we analyze the complexity of the proposed algorithm for solving the SMP.

We first look at the space complexity. One can easily see that the space complexity of Algorithm 1 is $\mathcal{O}(N_t^3)$ space since only a $N_t \times (N_t + 1)$ matrix should be stored. Similarly, the space complexity of Algorithm 2 is also $\mathcal{O}(N_t^3)$ space. The space complexities of Cholesky factorization (see (3)), the LLL reduction and saving \mathbf{Z} (see (6)) are $\mathcal{O}(N_t^3)$ space, thus the space complexity of the whole algorithm for solving the SMP is $\mathcal{O}(N_t^3)$ space.

In the following, we investigate the time complexity, in terms of flops, of the whole algorithm. Since the complexity of the Cholesky factorization, the expected complexity LLL reduction (when $1/4 < \delta < 1$) (see, e.g., [34]) and the complexity of computing \mathbf{ZB}^* (after finding a solution \mathbf{B}^* of the RSMP, a solution \mathbf{A}^* of the SMP can be obtained by setting $\mathbf{A}^* = \mathbf{ZB}^*$) are polynomial, and the complexity of Algorithm 2 is exponential, so the complexity of the whole algorithm is dominated by Algorithm 2.

In the sequel, we study the time complexity of Algorithm 2, which is also the complexity of the whole algorithm for solving the SMP by the above analysis.

From Algorithm 2, one can see that its complexity, denoted by $C(N_t)$, consists of two parts, the complexity of finding and updating integer vector \mathbf{b} satisfying (8), and that of updating \mathbf{B} whenever a nonzero integer vector \mathbf{b} is obtained, which are respectively denoted by $C_1(N_t)$ and $C_2(N_t)$. Then,

$$C(N_t) = C_1(N_t) + C_2(N_t).$$

Let $N_k(N_t)$ and f_k , $1 \leq k \leq N_t$, respectively be the number of nodes searched by the Schnorr-Euchner enumeration algorithm and the number of elementary operations, i.e., additions, subtractions, multiplications and divisions, that the enumeration performs for each visited node in the k -th level. By [35, Sec. IV-B], $f_k = \mathcal{O}(k)$, thus,

$$\begin{aligned} C_1(N_t) &= \sum_{k=1}^{N_t} N_k(N_t) f_k \leq \sum_{k=1}^{N_t} N_1(N_t) \mathcal{O}(k) \\ &= \mathcal{O}(N_t^2 N_1(N_t)). \end{aligned}$$

Since the number of times that \mathbf{B} needs to be updated is $N_1(N_t)$, and by Algorithm 1, each updating costs $\mathcal{O}(N_t^3)$ flops, so we obtain

$$C_2(N_t) = N_1(N_t) \mathcal{O}(N_t^3) = \mathcal{O}(N_t^3 N_1(N_t)).$$

By the aforementioned three equations, we have

$$C(N_t) = \mathcal{O}(N_t^3 N_1(N_t)). \quad (19)$$

To compute $C(N_t)$, we need to know $N_1(N_t)$, but unfortunately, exactly computing $N_1(N_t)$ is very difficult if it is not impossible. However, from [36], [37], [17] and [35], the expected value of $N_1(N_t)$, i.e., $E[N_1(N_t)]$, is proportional to

$$\frac{\pi^{N_t/2}}{\Gamma(N_t/2 + 1)} \beta^{N_t},$$

where $\beta = \|\bar{\mathbf{R}}\mathbf{b}_n\|_2$ (see the first sentence after (15)). Note that the above strategy has also been employed in [8] to analyze the complexity of the algorithm.

To compare the complexity of our proposed algorithm with that in [8], we make the same assumption as that in [8] on \mathbf{R} (see (3)), i.e., assume \mathbf{R} is the R-factor of the QR factorization of a matrix whose entries independent and identically follow the standard Gaussian distribution. Noting that the initial \mathbf{B} is a permutation of the $N_t \times N_t$ matrix, leading to $E[\|\mathbf{R}\mathbf{b}_k\|_2] = \sqrt{N_t}$ for $1 \leq k \leq N_t$, thus $E[\|\bar{\mathbf{R}}\mathbf{b}_n\|_2] \leq \sqrt{N_t}$. Hence,

$$E[N_1(N_t)] = \mathcal{O}\left(\frac{\pi^{N_t/2}}{\Gamma(N_t/2 + 1)} N_t^{N_t/2}\right). \quad (20)$$

By Stirling's approximation and the fact that $\Gamma(n+1) = n!$ for any positive integers n , we obtain

$$\frac{\pi^{N_t/2}}{\Gamma(N_t/2 + 1)} \approx \frac{1}{\sqrt{N_t \pi}} \left(\frac{2\pi e}{N_t}\right)^{N_t/2}.$$

Hence,

$$E[N_1(N_t)] \approx \mathcal{O}\left(\frac{1}{\sqrt{N_t}} (2\pi e)^{N_t/2}\right).$$

Combining with (19) yields,

$$E[C(N_t)] = \mathcal{O}(N_t^{5/2} (2\pi e)^{N_t/2}). \quad (21)$$

From (21), one can see that the expected complexity of the whole algorithm for solving the SMP is exponential in N_t .

B. Comparison of the complexity of the proposed method with that in [8]

In this subsection, we compare the complexity of the proposed algorithm with the one, i.e., Algorithm 2, for solving the real SMP in [8].

Note that two algorithms, which are respectively for solving real and complex SMPs, have been proposed in [8]. In this paper, we only developed an algorithm for solving real SMPs since a complex algorithm can be similarly developed and we omit its details due to the limitation of spaces.

By [8, Alg. 2], one can see that its space complexity is also $\mathcal{O}(N_t^3)$ space. So it has the same space complexity with the proposed one.

In the following, we examine their time complexities. By [8, eqs. (15) and (18)], the complexity of [8, Alg. 2] is

$$\mathcal{O}\left(N_t^4 \frac{\pi^{N_t/2}}{\Gamma(N_t/2 + 1)} N_t^{N_t/2}\right).$$

While, by (19) and (20), the complexity of the proposed algorithm is

$$\mathcal{O}\left(N_t^3 \frac{\pi^{N_t/2}}{\Gamma(N_t/2 + 1)} N_t^{N_t/2}\right).$$

Thus, the complexity of the new algorithm is an order of magnitude smaller than that of [8, Alg. 2] with respect to N_t .

In fact, one can use the following to explain the above result. From Sec. IV-B we know that [8, Alg. 2] needs to solve one SVP and $(N_t - 1)$ subspace avoiding problems. The complexity of solving a subspace avoiding problem is around $\mathcal{O}(N_t^3)$ times of that of solving a SVP, thus the total complexity is around $\mathcal{O}(N_t^4)$ times of that of solving a SVP. In contrast, the complexity of the new algorithm is around $\mathcal{O}(N_t^3)$ times of that of solving a SVP. Hence, the complexity of the new algorithm is an order of magnitude smaller than that of [8, Alg. 2] with respect to N_t .

VII. NUMERICAL RESULTS

This section presents simulation results to compare our proposed algorithm (denoted by “New Alg. ”) with the two optimal algorithms in [7] and [8] (denoted by “WC” and “DKWZ”, respectively). The average achievable rates and CPU time over 2000 random samples are reported.

All of the following simulations were performed on Matlab 2013a on the same desktop computer with Intel(R) Xeon(R) CPU E5-1603 v4 working at 2.80 GHZ.

Figure 2 shows the average achievable rates for the three algorithms with $N_t = 2, 4$. Figure 3 shows the average CPU time for the three algorithms with $N_t = 2$. Since “WC” is time consuming when $N_t = 4$, we display the the average CPU time for these algorithms in Table I.

Figures 4 and 5 respectively show average achievable rates and CPU time for “New Alg. ” and “DKWZ” with $P = 1, 10, 20$ dB and $N_t = 6 : 2 : 20$. We did not compare these with [7], which is clearly too slow.

Figures 2 and 4 indicate that the average rates for the three algorithms are exactly the same, and this is because all of them find optimal coefficient matrix \mathbf{A} .

From Table 3, Figures 3 and 5, one can see that the new algorithm is most efficient all the time, and “DKWZ” is faster than “WC” for $N_t = 4 : 2 : 20$.

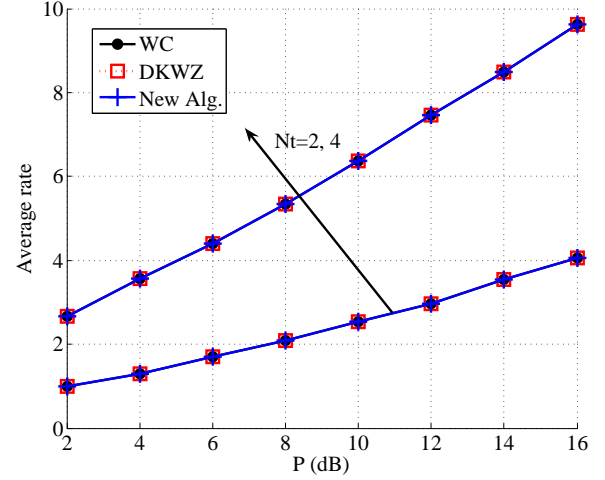


Fig. 2. Average rates over 2000 realizations for $N_t = 2, 4$

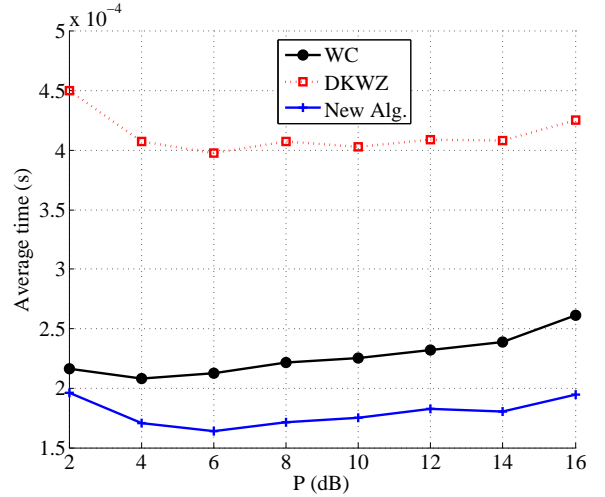


Fig. 3. Average CPU time over 2000 realizations for $N_t = 2$

TABLE I
AVERAGE CPU TIME OVER 2000 REALIZATIONS WITH $N_t = 4$

P (dB) \ Alg.	WC	DKWZ	New Alg.
2	0.0039	0.0022	0.00073
4	0.0201	0.0023	0.00078
6	0.2426	0.0023	0.00078
8	0.1827	0.0023	0.00080
10	1.810	0.0023	0.00085
12	13.32	0.0022	0.00081
14	26.23	0.0023	0.00089
16	32.69	0.0023	0.00089

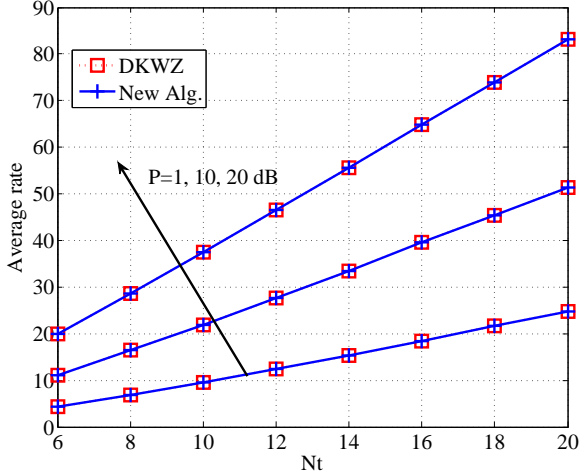


Fig. 4. Average rates over 2000 realizations for $P = 1, 10, 20$ dB

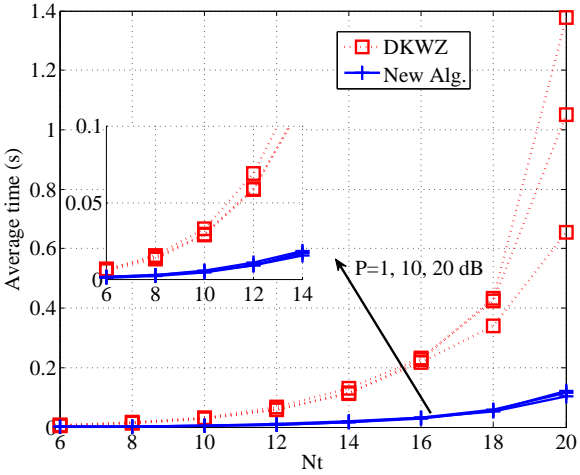


Fig. 5. Average CPU time over 2000 realizations for $P = 1, 10, 20$ dB

VIII. CONCLUSIONS

In this paper, we developed an efficient algorithm to find the optimal integer coefficient matrix which maximizes the achievable rate of the IF linear receiver. Different from the algorithm in [8] which forms the optimal matrix column by column, our algorithm initialized with a suboptimal matrix, which is then updated by a novel and efficient algorithm during the process of an improved version of sphere decoding until the optimal matrix is obtained. We have theoretically showed that the proposed algorithm is indeed an optimal algorithm. Theoretical complexity analysis showed that the complexity of

the new algorithm in big-O notation is an order of magnitude smaller, with respect to the dimension of the model matrix, than that of the fastest existing optimal algorithm which was proposed in [8]. Simulation results have also been given to confirm the efficiency of our algorithm.

APPENDIX A PROOF OF THEOREM 1

We prove Theorem 1 in this section. Clearly, if $i = n - 1$, then by the assumption, $\tilde{\mathbf{B}}_{[\setminus(n+1)]} = \tilde{\mathbf{B}}_{[1,n]}$ is invertible, i.e., the theorem holds. Thus, to show the theorem it suffices to show it still holds for $0 \leq i \leq n - 2$. We will prove it by contradiction. Specifically, we will show that if $\mathbf{B}_{[j]}$ is not invertible for all $i + 2 \leq j \leq n + 1$, then $\tilde{\mathbf{B}}_{[1,i+1]}$ is not full column rank which is contradict with the assumption. Before presenting the detailed proof, we show the following Claim.

Claim 1. Suppose that both $\tilde{\mathbf{B}}_{[k]}$ and $\tilde{\mathbf{B}}_{[\ell]}$ are not invertible for $i + 2 \leq k \neq \ell \leq n + 1$ with $0 \leq i \leq n - 2$, then there exist $f_j \in \mathbb{R}, 1 \leq j \neq i + 1, k, \ell \leq n + 1$ such that

$$\tilde{\mathbf{b}}_{i+1} = \sum_{j=1, j \neq i+1, k, \ell}^{n+1} f_j \tilde{\mathbf{b}}_j.$$

Proof of Claim 1. Since both $\tilde{\mathbf{B}}_{[k]}$ and $\tilde{\mathbf{B}}_{[\ell]}$ are not invertible, there exist $c_s \in \mathbb{R}, 1 \leq s \neq i + 1, k \leq n + 1$ and $d_t \in \mathbb{R}, 1 \leq t \neq i + 1, \ell \leq n + 1$ such that

$$\tilde{\mathbf{b}}_{i+1} = \sum_{j=1, j \neq i+1, k}^{n+1} c_j \tilde{\mathbf{b}}_j \text{ and } \tilde{\mathbf{b}}_{i+1} = \sum_{j=1, j \neq i+1, \ell}^{n+1} d_j \tilde{\mathbf{b}}_j. \quad (22)$$

By (22), one can immediately obtain that

$$\sum_{j=1, j \neq i+1, k}^{n+1} c_j \tilde{\mathbf{b}}_j = \sum_{j=1, j \neq i+1, \ell}^{n+1} d_j \tilde{\mathbf{b}}_j.$$

Without loss of generality, we assume $k < \ell$, then the aforementioned equation can be rewritten as (note that we assume $\sum_{j=i_1}^{i_2} \cdot = 0$ whenever $i_2 < i_1$ in the sequel)

$$\begin{aligned} \sum_{j=1, j \neq i+1}^{k-1} (c_j - d_j) \tilde{\mathbf{b}}_j - d_k \tilde{\mathbf{b}}_k + \sum_{j=k+1}^{\ell-1} (c_j - d_j) \tilde{\mathbf{b}}_j \\ + c_\ell \tilde{\mathbf{b}}_\ell + \sum_{j=\ell+1}^{n+1} (c_j - d_j) \tilde{\mathbf{b}}_j = \mathbf{0}. \end{aligned}$$

Since $\mathbf{B} = \tilde{\mathbf{B}}_{[\setminus(i+1)]}$ is invertible, thus one can obtain from the above equation that

$$d_k = c_\ell = 0.$$

From the aforementioned equation and (22), one can see that the Claim holds.

In the following, we use Claim 1 to prove Theorem 1.

Proof of Theorem 1. By the analysis given in the beginning of this Appendix, we only need to show the theorem still holds for $0 \leq i \leq n-2$. We prove it by contradiction. Suppose that $\mathbf{B}_{[n,j]}$ is not invertible for all $i+2 \leq j \leq n+1$, then by Claim 1 with $k = i+2, \ell = i+3$, there exist $f_s \in \mathbb{R}, 1 \leq s \neq i+1, i+2, i+3 \leq n+1$ such that

$$\tilde{\mathbf{b}}_{i+1} = \sum_{s=1, s \neq i+1, i+2, i+3}^{n+1} f_s \tilde{\mathbf{b}}_s. \quad (23)$$

If $i+3 = n+1$, i.e., $i = n-2$, from (23), one can see that $\tilde{\mathbf{b}}_{i+1} = \sum_{s=1}^i f_s \tilde{\mathbf{b}}_s$, which implies that $\tilde{\mathbf{B}}_{[1, i+1]}$ is not full column rank. Clearly, this contradicts with the assumption, thus the assumption that $\mathbf{B}_{[n,j]}$ is not invertible for all $i+2 \leq j \leq n+1$ is not correct. In other words, the theorem holds in this case.

In the following, we assume $i < n-2$ and prove that $f_j = 0$ for any given $i+4 \leq j \leq n+1$. Since $\tilde{\mathbf{B}}_{[n,j]}$ is not invertible for all $i+2 \leq j \leq n+1$, by Claim 1 with $k = i+2, \ell = j$, there exist $\tilde{f}_t \in \mathbb{R}, 1 \leq t \neq i+1, j \leq n+1$ such that

$$\tilde{\mathbf{b}}_{i+1} = \sum_{t=1, t \neq i+1, i+2, j}^{n+1} \tilde{f}_t \tilde{\mathbf{b}}_t.$$

Combing with (23), one can obtain

$$\sum_{s=1, s \neq i+1, i+2, i+3}^{n+1} f_s \tilde{\mathbf{b}}_s = \sum_{t=1, t \neq i+1, i+2, j}^{n+1} \tilde{f}_t \tilde{\mathbf{b}}_t,$$

which can be rewritten as

$$\begin{aligned} \sum_{s=1}^i (f_s - \tilde{f}_s) \tilde{\mathbf{b}}_s - \tilde{f}_{i+3} \tilde{\mathbf{b}}_{i+3} + \sum_{s=i+4}^{j-1} (f_s - \tilde{f}_s) \tilde{\mathbf{b}}_s \\ + f_j \tilde{\mathbf{b}}_j + \sum_{s=j+1}^{n+1} (f_s - \tilde{f}_s) \tilde{\mathbf{b}}_s = \mathbf{0}. \end{aligned}$$

Since $\mathbf{B} = \tilde{\mathbf{B}}_{[n, i+1]}$ is invertible, the aforementioned equation implies $f_j = 0$. Since $f_j = 0$ for any given $i+4 \leq j \leq n+1$, by (23), $\tilde{\mathbf{b}}_{i+1}$ is a linear combination of vectors $\tilde{\mathbf{b}}_1, \dots, \tilde{\mathbf{b}}_i$, i.e., $\tilde{\mathbf{B}}_{[1, i+1]}$ is not full column rank. Again, this contradicts with the assumption, thus the theorem also holds in this case. \square

REFERENCES

- [1] A. Lenstra, H. Lenstra, and L. Lovász, "Factoring polynomials with rational coefficients," *Math. Ann.*, vol. 261, no. 4, pp. 515–534, 1982.
- [2] D. Wübben, R. Bohnke, J. Rinas, V. Kuhn, and K. Kammeyer, "Efficient algorithm for decoding layered space-time codes," *Electron. Lett.*, vol. 37, no. 22, pp. 1348–1350, 2001.
- [3] G. J. Foschini, G. D. Golden, R. A. Valenzuela, and P. W. Wolniansky, "Simplified processing for high spectral efficiency wireless communication employing multi-element arrays," *IEEE J. Sel. Areas Commun.*, vol. 17, no. 11, pp. 1841–1852, 1999.
- [4] X.-W. Chang, J. Wen, and X. Xie, "Effects of the LLL reduction on the success probability of the babai point and on the complexity of sphere decoding," *IEEE Trans. Inf. Theory*, vol. 59, no. 8, pp. 4915–4926, 2013.
- [5] J. Zhan, B. Nazer, U. Erez, and M. Gastpar, "Integer-forcing linear receivers: A new low-complexity MIMO architecture," in *2010 IEEE 72nd VTC 2010-Fall*. IEEE, 2010, pp. 1–5.
- [6] —, "Integer-forcing linear receivers," *IEEE Trans. Inf. Theory*, vol. 60, no. 12, pp. 7661–7685, 2014.
- [7] L. Wei and W. Chen, "Integer-forcing linear receiver design over MIMO channels," in *GLOBECOM*, 2012, pp. 3560–3565.
- [8] L. Ding, K. Kansanen, Y. Wang, and J. Zhang, "Exact SMP algorithms for integer-forcing linear MIMO receivers," *IEEE Trans. Wireless Commun.*, vol. 14, no. 12, pp. 6955–6966, 2015.
- [9] A. Sakzad, J. Harshan, and E. Viterbo, "Integer-forcing MIMO linear receivers based on lattice reduction," *IEEE Trans. Wireless Commun.*, vol. 12, no. 10, pp. 4905–4915, 2013.
- [10] L. Wei and W. Chen, "Integer-forcing linear receiver design with slowest descent method," *IEEE Trans. Wireless Commun.*, vol. 12, no. 6, pp. 2788–2796, 2013.
- [11] A. Sakzad and E. Viterbo, "Full diversity unitary precoded integer-forcing," *IEEE Trans. Wireless Commun.*, vol. 14, no. 8, pp. 4316–4327, 2015.
- [12] O. Ordentlich and U. Erez, "Precoded integer-forcing universally achieves the mimo capacity to within a constant gap," *IEEE Trans. Inf. Theory*, vol. 61, no. 1, pp. 323–340, 2015.
- [13] G. Golub and C. Van Loan, "Matrix computations, 4th," *Johns Hopkins*, 2013.
- [14] C. Feng, D. Silva, and F. R. Kschischang, "An algebraic approach to physical-layer network coding," *IEEE Trans. Inf. Theory*, vol. 59, no. 11, pp. 7576–7596, 2013.
- [15] B. Nazer, V. Cadambe, V. Ntranos, and G. Caire, "Expanding the compute-and-forward framework: Unequal powers, signal levels, and multiple linear combinations," *IEEE Trans. Inf. Theory*, vol. 62, no. 9, pp. 4879–4909, Sept. 2016.
- [16] W. H. Mow, "Maximum likelihood sequence estimation from the lattice viewpoint," *IEEE Trans. Inf. Theory*, vol. 40, no. 5, pp. 1594–1600, 1994.
- [17] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Trans. Inf. Theory*, vol. 48, no. 8, pp. 2201–2214, 2002.
- [18] M. O. Damen, H. E. Gamal, and G. Caire, "On maximum likelihood detection and the search for the closest lattice point," *IEEE Trans. Inf. Theory*, vol. 49, no. 10, pp. 2389–2402, 2003.
- [19] D. Micciancio and O. Regev, *Lattice-Based Cryptography*. Bernstein, D. J. and Buchmann, J. (eds.), Berlin: Springer Verlag, 2008.
- [20] G. Hanrot and D. Stehlé, "Improved analysis of kannan's shortest lattice vector algorithm," in *Proceedings of the 27th annual international cryptology conference on Advances in cryptology*. Springer-Verlag, 2007, pp. 170–186.
- [21] G. Hanrot, X. Xavier Pujol, and D. Stehlé, "Algorithms for the shortest and closest lattice vector problems," in *IWCC'11 Proceedings of the Third international conference on Coding and cryptology*. Springer-Verlag Berlin, Heidelberg, 2011, pp. 159–190.
- [22] C. Schnorr and M. Euchner, "Lattice basis reduction: improved practical algorithms and solving subset sum problems," *Math Program*, vol. 66, pp. 181–191, 1994.
- [23] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," *Math. Comput.*, vol. 44, no. 170, pp. 463–471, 1985.

- [24] X.-W. Chang and Q. Han, "Solving box-constrained integer least squares problems," *IEEE Trans. Wireless Commun.*, vol. 7, no. 1, pp. 277–287, 2008.
- [25] T. Cui and C. Tellambura, "Approximate ml detection for mimo systems using multistage sphere decoding," *IEEE Signal Process. Lett.*, vol. 12, no. 3, 2005.
- [26] A. Ghaderipoor and C. Tellambura, "A statistical pruning strategy for schnorr-euchner sphere decoding," *IEEE Wireless Commun. Lett.*, vol. 12, no. 2, pp. 121–123, 2008.
- [27] T. Cui, S. Han, and C. Tellambura, "Probability-distribution-based node pruning for sphere decoding," *IEEE Trans. Veh. Technol.*, vol. 62, no. 4, pp. 1586–1596, 2013.
- [28] J. Wen, B. Zhou, W. H. Mow, and X.-W. Chang, "An efficient algorithm for optimally solving a shortest vector problem in compute-and-forward design," *IEEE Trans. Wireless Commun.*, vol. 15, no. 10, pp. 6541–6555, 2016.
- [29] J. Blömer and S. Naewe, "Sampling methods for shortest vectors, closest vectors and successive minima," in *International Colloquium on Automata, Languages, and Programming*. Springer, 2007, pp. 65–77.
- [30] —, "Sampling methods for shortest vectors, closest vectors and successive minima," *IEEE Trans. Commun.*, vol. 410, no. 18, pp. 1648–1665, 2009.
- [31] S. Banerjee and A. Roy, *Linear algebra and matrix analysis for statistics*. CRC Press, 2014.
- [32] H. Anton and C. Rorres, *Elementary Linear Algebra: Applications Version*. Wiley Global Education, Oct. 2013.
- [33] J. Wen and X.-W. Chang, "A new KZ reduction algorithm," *In preparation*, 2016.
- [34] C. Ling, W. Mow, and N. Howgrave-Graham, "Reduced and fixed-complexity variants of the LLL algorithm for communications," *IEEE Trans. Commun.*, vol. 61, no. 3, pp. 1040–1050, 2013.
- [35] B. Hassibi and H. Vikalo, "On the sphere-decoding algorithm I. Expected complexity," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 2806–2818, 2005.
- [36] P. M. Gruber and J. M. Wills, Eds., *Handbook of convex geometry*. North-Holland, Amsterdam, 1993.
- [37] A. Banihashemi and A. K. Khandani, "On the complexity of decoding lattices using the korkin-zolotarev reduced basis," *IEEE Trans. Inf. Theory*, vol. 44, no. 1, pp. 162–171, 1998.